Sequenced Data Acquisition

Fermilab
Beams Division
Accelerator Controls Department

Kevin Cahill

Version 1.1
August 6, 2002

Architecture

Components

Sequenced Data Acquisition (SDA) is a term that describes a collection of several components working together within a Data Acquisition Engine to collect, store, and describe rule-based data acquisition. Those components include an Open Access Client front-end, a data logger, report writers, and a series of data acquisition jobs.

Data Acquisition Job

By definition, a job with source, disposition, item and event job components of AcceleratorSource, SavedDataDisposition, SDADataItem, and EmptyDataEvent respectively define a SDA job. SDADataItem is an abstract class. ColliderShotItem and PbarTransferShotItem are examples of classes that extend SDADataItem.

SavedDataDisposition

SavedDataDisposition supports multiple collections of scalar and snapshot data by tagging each datum with an owner, file alias, file index, collection alias, set alias, and collection index. Examples of owners include “SaveFile”, “ColliderShot”, and “PbarTransferShot”. File alias is a file, shot, or store number. File index is a unique, incrementing number across all save files. Collection alias is a case number describing, for example, proton injection. Set alias is a set number within the case describing the proton bunch number for example. Collection index is a unique, incrementing number across each save set within a single save file.

SDADataItem

SDADataItem manages the data collection devices and rules. It provides a test environment for testing SDA collection. It provides default and abstract methods for SDA.

EmptyDataEvent

The EmptyDataEvent simply specifies that the data collection event(s) are not found in the job’s DataEvent, but in its DataItem.

SequencedScheduler

When a SDA job is instantiated on an engine the SequencedScheduler class manages the creation of other data acquisition jobs to accomplish the SDA.

Overview of a Collider Shot

DUE05, a data acquisition engine, is started. Open Access Clients on DUE05 are instantiated including CBSHOT, Collider’s SDA, and MCRVCR, Collider’s video tape recording Open Access Client. The data logger CBSDA is also started.

CBSHOT’s devices are downloaded as part of the Open Access Client (oac) architecture. When download completes, CBSHOT instantiates and initializes a ColliderShotItem defining an operational or test mode and the naming convention for control, reporting, and diagnostic devices owned by the front-end client.

ColliderShotItem reads the database for device and data collection rules and begins an internal MonitorStore to control the deployment of the SDA. If a testing mode is enabled, it is started at this time.

Each SDA Open Access Client provides devices to identify store number and disabling and shutdown devices. If the SDA is enabled, the MonitorStore within the SDADataItem starts the overall SDA. It continues to monitor for shutdown events and new stores, stopping and starting the controlling SDA job as needed. Device V:SDAEND will cause the Collider SDA to close out when a software state transition is raised. Device

T:STORE will cause the present Collider SDA to close and a new SDA to initialize when the store number changes.

The SequencedScheduler obtains a list of SDACollection objects from the SDADataItem. It begins to monitor the disabling, shutdown, and mode event devices for the SDA. V:SDAENB is the enabling/disabling device for Collider shots. If the SDA is disabled, new cases will not be armed. Device V:TEVMOD defines the Collider SDA state. The mode event describes collider shot modes such as collision and proton only transfers. The scheduler sets several oac devices to define to the control system the current SDA shot number, file index, collection case, set number, collection index, collection arming time and set collection time. For Collider SDA, those devices are C:FILE, C:FINDEX, C:CASE, C:SET, C:CINDEX, C:ARMTIM, and C:SETTIM.

The SequencedScheduler obtains a list of SDACollection objects from the SDADataItem that define the data acquisition specifications of the SDA. Collider SDA defines 28 cases. Examples include Proton Injection tune up, Inject Protons, Inject Pbars, Acceleration, and HEP. Each case supports two arming, disarming, and ending events. Arming enables the monitoring of set collection events or starts a collection for the null set collection event. Disarming disables the monitoring of set collection events. For Collider SDA only one of each is defined, and all of these events are software state transition events. Each case may define two set collection events, and each case may specify whether the case can be armed more than once within a single SDA shot, if the case may run concurrently with other cases, and if the set events may cause multiple collections within the case. Collider SDA defines a single set collection event for each case, and set arming events include software state transition devices and Tevatron clock events as well as the null event for cases without multiple set collections. Most Collider SDA disarming and ending events are defined to as a software state transition to another case plus a time delay.

The SequencedScheduler monitors all SDACollections' arming and disarming events. When an arming event is detected, active cases not supporting concurrent cases are disarmed. The set collection events of the armed case if existing are monitored, otherwise a set collection event is declared immediately. The firing of a set collection arm event will create jobs to collect scalar, snapshot, and fast time plot data after terminating previous set collection jobs for the same case. Each scalar, snapshot, and fast time plot collection specification within the collection may have unique collection specifications. Scalar collection on a Tevatron clock event plus delay or on a software state transition plus delay, or on an absolute time delay after the set collection event are supported. Snapshot collection supports the entire range of the snapshot protocol. Any control system channel may be specified for scalar and snapshot collection regardless of the support of the destination front-end as the data acquisition engine internally supports front-ends lacking protocol coverage. Fast time plot collection may be used to collect a larger number of points, slower rates, and longer durations than snapshot collection.

SavedDataDisposition receives the scalar, snapshot, and fast time plot data collected by the set collection jobs into a SetDisposition inner class created by SequencedScheduler for each set collection. The data is stored in the Sybase database as raw data suitable for retrieval by any Save/Restore capable program. Scalar data may be scaled and stored in the database as scaled data (this feature is turned off) as are regular Save/Restore saves by data acquisition engines. Scaled, scalar data is also logged in the oac's associated data logger. For ColliderShot SDA this logger is CBSDA. Beam position monitoring data is filtered from the incoming data and passed to a VMS job that returns scaled beam position data for database and data logger storage.

Open Access Client Devices supporting SDA

Each SDA Open Access Client instance defines a name prefix for identifying SDA devices. ColliderShot has a prefix of "C:"
The Open Access Client supports several devices reflecting SDA's status.

C:FILE	file alias (T:STORE)
C:FINDEX	file index (unique, auto-incrementing)
C:MODE	mode (state value of V:TEVMOD)
C:CASE	case alias (e.g. 4; ProtonInjection, value of V:CLDRST)
C :ARMTIM	case arm time (seconds since start of shot)
C :SET	set alias (e.g. V:NXBNCH value for case 4)
C:SETTIM	set time (seconds since start of shot)
C:DISTIM	case disarm time (seconds since start of shot)
C:ENDTIM	case end time (seconds since start of shot)
C:DAQTIM	set scalar complete time (seconds since start of shot)
C:SNPTIM	set snap complete time (seconds since start of shot)
C:FTPTIM	set FTP complete time (seconds since start of shot)
C:GOOD	set from 'good' shot when 1, else 0
C:REPEAT	set is store over when 1, else 0

SDA Data Loggers

SDA data loggers are instances of DataLoggerArchiveDisposition, storing data in multiple MySQL databases where each month's data resides in a database. All of the scalable, scalar data is stored in the database forever. SDA data loggers are unusual in saving scalar data under a variety of names to aid in retrieval.

Initially, SDA logged devices as it wrote database entries. It now does most logging after a shot completes. To aid in matching timestamps, all logged entries are time-stamped with the start time of the set collection. Further, for each set collection, identifier devices are logged to aid in determining the file, case, and set.

Imagine a time period covering 10 shots (1670-1680) and an interest in the device element C:FBIPNG[36](last bunch) in the proton injection case(4):

Collecting C:FBIPNG[36] over t1/t2 will retrieve all logged values across all cases.

Collecting C:FBIPNG[36],4 will retrieve all logged values across the proton injection case.

Collecting C:FBIPNG[36],4,36 will retrieve all logged values from set 36 of proton injections.

Collecting C:FBIPNG[36],F,4,36 will retrieve all logged values from set 36 of proton injection of 'good' shots and will not include store overs.

Collecting C:CASE or C:CASE,4 or C:CASE,4,35 or C:CASE,F,4,35 will retrieve logged case values that will include timestamps that match the above collection of C:FBIPNG as C:FILE, C:FINDEX, C:MODE, C:CASE, C:CINDEX, C:SET, and C:SETTIM are logged on each set collection. G:GOOD and C:REPEAT are logged on each set collection as booleans describing 'good' shots and store overs respectively.

Collecting C:FILE over t1/t2 will return a logged file number for each collection (many more than 10).

The devices C:UFILE, C:UFINDX, C:UMODE, C:UCASE, C:UCINDX, C:USET, C:USETTM are the equivalent 'unique' devices that will be logged but once, so C:UFILE would return but 10 points for this time period. Again, the unique devices are also logged on 'good' shots so C:FILE,F may return less than 10 points.

Collider SDA Open Access Client devices

The Collider Open Access Client monitors collected data and produces reports. Each of the cells of the reports represent a device that may be viewed on a parameter page, SDA_Viewer, or a data logger plotter. The efficiency report and its mapping to devices are shown in the next two images:

Shot Summary, store 1613, Wed Jul 31 20:25:56 CDT 2002, Initial Stack Size = 87.58749999999999 E10

	Proton Intensity E9	Step Efficiency %	Cumulative Efficiency %	Pbar Intensity E9	Step Efficiency %	Cumulative Efficiency %
Accumulator				700.00		
AP3				702.76	100.39	100.39
AP1				672.96	95.76	96.14
P2				655.49	97.40	93.64
P1				819.65	125.04	117.09
MI Inj				585.36	71.42	83.62
MI 8Gev				680.78	116.30	97.25
MI 150Gev				716.26	105.21	102.32
Coalescing				456.45	63.73	65.21
Inject Protons	86862.38					
Pbar Injection porch	8864.53	10.21				
Inject Pbars	6812.42	76.85		415.76	91.09	59.39
Before Ramp	6791.77	99.70		352.32	84.74	50.33
Flattop	6363.15	93.69		318.79	90.48	45.54
Squeeze	6296.40	98.95		308.72	96.84	44.10
Initiate Collisions	6299.69	100.05		305.25	98.88	43.61
Remove Halo	6156.92	97.73		301.64	98.82	43.09
HEP	6133.87	99.63		305.43	101.26	43.63
Initial Luminosity	13.70	CDF		12.99	DZero	
Shot Setup Time	136.56	min				

Shot Summary, mapping

	Proton Intensity	Step Efficiency	Cumulative Efficiency	Pbar Intensity	Step Efficiency	Cumulative Efficiency
Accumulator				A:AINT06		
AP3				A:AINTA3	A:AEFFA3	A:ACEFA3
AP1				A:AINTA1	A:AEFFA1	A:ACEFA1
P2				A:AINTP2	A:AEFFP2	A:ACEFP2
P1				A:AINTP1	A:AEFFP1	A:ACEFP1
MI Inj				A:AINTMI	A:AEFFMI	A:ACEFMI
MI 8Gev				I:AINT8G	I:AEFF8G	I:ACEFFT
MI 150Gev				I:AINTFT	I:AEFFFT	I:ACEFFT
Coalescing	I:PINT04			I:AINT06	I:AEFF04	I:ACEF04
Inject Protons	C:PINT04	C:PEFF04	C:PCEF04			
Pbar Injection porch	C:PINT05	C:PEFF05	C:PCEF05			
Inject Pbars	C:PINT06	C:PEFF06	C:PCEF06	C:AINT06	C:AEFF06	C:ACEF06
Before Ramp	C:PINT08	C:PEFF08	C:PCEF08	C:AINT08	C:AEFF08	C:ACEF08
Flattop	C:PINT10	C:PEFF10	C:PCEF10	C:AINT10	C:AEFF10	C:ACEF10
Squeeze	C:PINT11	C:PEFF11	C:PCEF11	C:AINT11	C:AEFF11	C:ACEF11
Initiate Collisions	C:PINT12	C:PEFF12	C:PCEF12	C:AINT12	C:AEFF12	C:ACEF12
Remove Halo	C:PINT13	C:PEFF13	C:PCEF13	C:AINT13	C:AEFF13	C:ACEF13
HEP	C:PINT14	C:PEFF14	C:PCEF14	C:AINT14	C:AEFF14	C:ACEF14
Initial Luminosity	C:ILUMB0			C:ILUMD0		
Initiate Collisions	C:SHTTIM					

The emittances report and its mapping to devices are shown in the next two images:

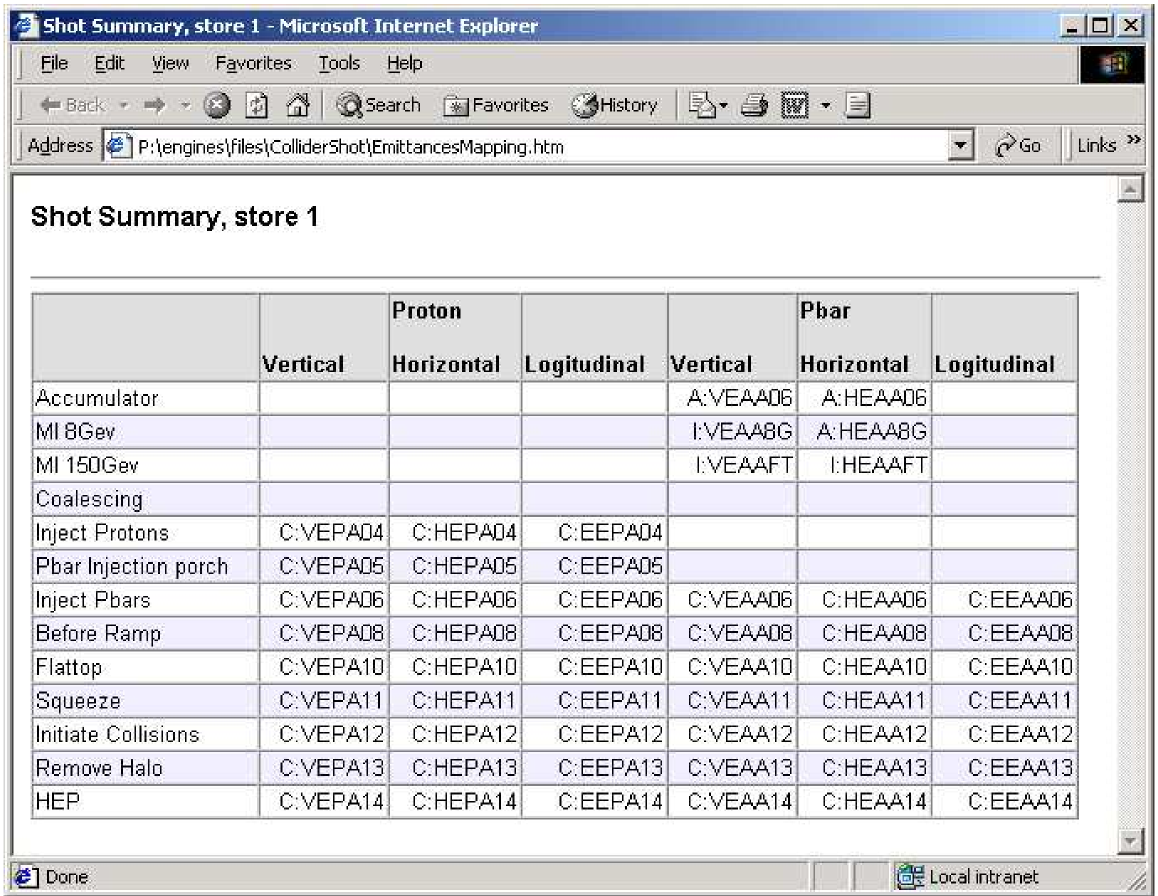
Shot Summary, store 1616, Fri Aug 02 04:36:37 CDT 2002, Initial Stack Size = 94.5875 E10 - Microsoft Internet Ex...

Address P:\engines\files\ColliderShot\ColliderShotEmittances_1616.html

Shot Summary, store 1616, Fri Aug 02 04:36:37 CDT 2002, Initial Stack Size = 94.5875 E10

	Proton			Pbar		
	Vertical	**Horizontal**	**Logitudinal**	**Vertical**	**Horizontal**	**Logitudinal**
Accumulator				5.20	5.97	4.30
MI 8Gev				4.94	6.37	0.28
MI 150Gev				21208.04	10.21	0.20
Coalescing						
Inject Protons	24.20	24.83	5.20			
Pbar Injection porch	24.34	27.15	5.37			
Inject Pbars	21.99	18.73	5.44	16.69	21.23	4.53
Before Ramp	21.99	18.73	5.44	15.00	19.50	4.79
Flattop						
Squeeze	21.76	20.14	1.25	18.07	18.53	1.06
Initiate Collisions	21.76	20.14	1.25	18.07	18.53	1.06
Remove Halo	22.79	19.04	1.31	17.67	20.12	1.09
HEP						
Initial Luminosity	20.09	CDF		18.52	DZero	

Done | Local intranet



Shot Summary, store 1

	Proton Vertical	 Horizontal	 Logitudinal	Pbar Vertical	 Horizontal	 Logitudinal
Accumulator				A:VEAA06	A:HEAA06	
MI 8Gev				I:VEAA8G	A:HEAA8G	
MI 150Gev				I:VEAAFT	I:HEAAFT	
Coalescing						
Inject Protons	C:VEPA04	C:HEPA04	C:EEPA04			
Pbar Injection porch	C:VEPA05	C:HEPA05	C:EEPA05			
Inject Pbars	C:VEPA06	C:HEPA06	C:EEPA06	C:VEAA06	C:HEAA06	C:EEAA06
Before Ramp	C:VEPA08	C:HEPA08	C:EEPA08	C:VEAA08	C:HEAA08	C:EEAA08
Flattop	C:VEPA10	C:HEPA10	C:EEPA10	C:VEAA10	C:HEAA10	C:EEAA10
Squeeze	C:VEPA11	C:HEPA11	C:EEPA11	C:VEAA11	C:HEAA11	C:EEAA11
Initiate Collisions	C:VEPA12	C:HEPA12	C:EEPA12	C:VEAA12	C:HEAA12	C:EEAA12
Remove Halo	C:VEPA13	C:HEPA13	C:EEPA13	C:VEAA13	C:HEAA13	C:EEAA13
HEP	C:VEPA14	C:HEPA14	C:EEPA14	C:VEAA14	C:HEAA14	C:EEAA14